**Word Count: 2,110**

### Integration and System Testing

In this report I will be writing about integration and system testing which is a type of testing which is conducted in the software development process. I will start off by writing about integration testing.

## Introduction

Integration and system testing is a type of software testing, this makes sure that tests such as the system and integration are done before releasing the product. Software testing has very strict set of rules and guidelines that it follows to make sure each individual part of the software is thoroughly checked before it is given the OK, this makes sure that there are no errors and that the software runs how it’s supposed to.

Integration and system testing is mainly done by a team who focuses only on the software testing phase in the system development life cycle.

In software testing each testing level build on from the previous level so it is important that the testing is done in the correct order, access to the information is passed on to the next level.

## Integration

Integration testing in the software testing model comes before system testing and after the unit testing has been done.

The way that integration testing works is by, getting the individual modules that have been through the unit testing phase and integrating each module into a group. The integration testing phase will make sure when the modules are being integrated together that any problems, for example errors or bugs, caused due to the integration of the modules are eliminated. Integration testing does not deal with the integration of the whole system but deals with the integration of a process in the system.

In the integration testing stage there are three things that are created, to ensure that the integration of the modules is successful and that it runs successfully as well, a test plan, test cases and test data is produced to effectively test that the integration is successful. Test data is normally used by test cases but I have mentioned each type below;

### Integration Test Plan

When producing a test plan it must include the following information to be effective,

A strategy to use when testing the integrated modules and how the tests will be conducted.

 What will be tested for example software features.

 What is the time scale and time management.

 Responsibilities, e.g. personnel.

 Testing pass and fail condition.

 Risk involved

 Approval from all important people involved.

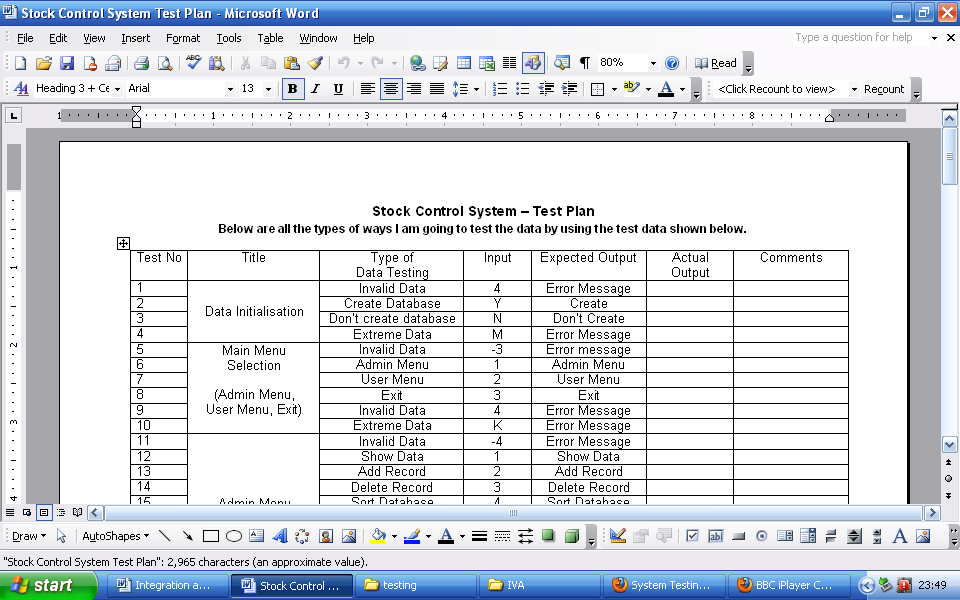
Above are just some of the most important information that is in the test plan but as a test plan is simply planning there is more information and points that can be included in the test plan by a company.

Most test plans are approved and worked on with the client so they may order some changes later on, so a test plan may have to include more information and it is best to get this approved so no problems are encountered later on.

### Integration Test Cases

Test cases is created to make sure that the output, of the integrated modules are producing the expected output and is working exactly how it is supposed to work. This is simply a way to spot any errors or bugs that might have been made in the integration phase. The tester will then work through the program and document all the data using the test case that was created, the test case will test all inputs and outputs in the integrated modules.

Below is a simple test case, this was created by myself for a program that I created in collage, I have also included some test data in the test case to show exactly how it works.



If you are dealing with a large application or program then there may be various test cases that might need to be created to test separate sections of the program. The various test cases are normally gathered together and referred to as test suites, which is a set of test cases.

### Integration Test Data

Test data is simply data that is used in order to test the actual program or the integrated modules. Test data would normally be used in a test case as this would be used to check the inputs and expected outputs.

I have included an example of what test data is in the test case example that was shown in the test case section above.

### Different types of Integration Testing

There are some different types of integration testing that can be conducted, below is a list of the different integration testing types;

**Big Bang**

The way this integration testing type works is, most or all of the modules are integrated together to form nearly a complete system. This is very similar to system testing as this basically has a whole system before starting the testing.

There are not many advantages of the big bang integration; the only advantage is that for a smaller system this will be the ideal integration testing technique.

The disadvantages is that you would have to wait for all the modules to be integrated in order to do big-bang testing so there will be quite a lot of delay. Any errors are identified at a very late stage and it is very hard to indentify the fault. It is very difficult to be sure that all testing has been done before product release.

**Top-Down testing**

This is where the highest level components are tested first and then step by step start working downwards (lower components). Top-down testing mainly requires for the testing team to separate what is important and what is least important, then the most important modules are worked on first. The top-down approach is similar to a binary tree you would start off by integrating the top level before slowly working your way down the tree and integrating all the components at that level.

The advantage to this way of testing is that if a prototype is released or shown then most of the main functionality will already be working. It is also easy to maintain the code and there will be better control in terms of errors so most of the errors would be taken out before going to the next stage of testing.

The disadvantage is that it is hard to test the lower level components using test data. The other thing is that lower level modules may not be tested as much as the upper level modules.

**Bottom-up testing**

This is the opposite to top-down where you test the lower level components and start testing your way upwards to the higher level components. The components will be separated into the level of importunacy and the least important modules will be worked on first, then slowly you would work your way up by integrating components at each level before moving upwards.

The advantage is that with this method you would be able to maintain code more easily and there is a more clear structure of how to do things.

The disadvantage is that when releasing a prototype you can not see a working prototype until nearly all the program has been completed so that may take a long time before this happens. There may be a lot of errors later on regarding the GUI and programming later on.

### Integration – Overall

Integration testing is best to be used in an iterative process as this way it will save time in the long run and will help when trying to keep to a budget.

The reason why a iterative process is the best for integration testing is simply because this allows for more feedback from the client, so if the client is involved with the project a lot it would be less likely in terms of having to make a lot of changes in the integration process, for example test plan.

The best integration testing type would be top-down as this is the fastest way for the integration process to be completed. There is however one problem that may cause time delay in the integration process that is the when using testing data to test the process if faults or errors or reported this will have to be documented, fixed and re-done so this will cause a delay in the time taken.

Integration testing again uses black box testing as all the integration stage is doing is checking for the correct outputs.

## System Testing

System testing is simply testing the system as a whole; it gets all the integrated modules of the various components from the integration testing phase and combines all the different parts into a system which is then tested. Testing is then done on the system as all the parts are now integrated into one system the testing phase will now have to be done on the system to check and remove any errors or bugs.

In the system testing process the system will be checked not only for errors but also to see if the system does what was intended, the system functionality and if it is what the end user expected.

There are various tests that need to be conducted again in the system testing which include;

* Test Plan
* Test Case
* Test Data

If the integration stage was done accurately then most of the test plan and test cases would already have been done and simple testing would only have to be done in order to ensure there are no bugs because this will be the final product.

As in the integration stage, the above steps would need to be re-done as now we have integrated all modules into one system so we have to check if this runs OK and that no errors are produced because all the modules are in one system.

### System Test Plan

The test plan will contain similar information to what was included in the integration testing, but would contain more information as this time we are not doing individual sections but whole systems.

### System Test Case

The test case would also have to change to test the whole system again to see if no errors turned up after combining into a whole system. The test case would include test data to test expected output.

### Different types of System Testing

There are loads of examples of system testing; below I will discuss some of the important types of systems testing that are done regularly,

* Usability testing – this is how well the user can access the different features in the system and how easy it is to use.
* GUI software testing – this is to check if graphically that the program looks how was intended and the GUI works as intended.
* Security testing – this would be to check if important information is secure and if there are certain access restriction that they work.
* Accessibility – how easy is it for various users including users with disability to use the system.
* Reliability testing – to check that the system works for long period of time and does not constantly crash.

Below is a full list that is shown on Wikipedia of all the different types of system testing that is available

[[1]](#footnote-1)

|  |
| --- |
| * GUI software testing * Usability testing * Performance testing * Compatibility testing * Error handling testing * Load testing * Volume testing * Stress testing * User help testing * Security testing * Scalability testing * Capacity testing * Sanity testing * Smoke testing * Exploratory testing * Ad hoc testing * Regression testing * Reliability testing * Recovery testing * Installation testing * Idempotency testing * Maintenance testing * Recovery testing * Accessibility |

### System Testing – Overall

System testing has a group of people who just deal with the testing side of the process, so this testing stage uses black box testing as the testing team only really deal with the output of the system and the documenting of any problems due to the output.

This is similar to integration testing in that the time taken to complete this testing is dependent on not many errors or bugs appearing in the testing phase, or this stage can take a long time to complete.

The reason why it is important to do system testing is that it can in the long run help in terms of staying within a budget and meeting a deadline.

In my opinion in the testing phases the more documentation that is produced in terms of documenting errors and producing detailed test plans the quicker it would be to finish the product with fewer errors or bugs. If these tests were not done then I believe that there would be more bugs or errors as well as damage to company reputation because the end user will keep complaining about system failure and negative views about the company would spread, via word of mouth.

## References

*G53QAT Presentation, Collin Higgins [online], Available at:*

[*http://www.cs.nott.ac.uk/~cah/G53QAT/G53QAT08pdf6up.pdf*](http://www.cs.nott.ac.uk/~cah/G53QAT/G53QAT08pdf6up.pdf) *[Accessed 20th November 2009]*

*Integration testing, Wikipedia [online], Available at:*

[*http://en.wikipedia.org/wiki/Integration\_testing*](http://en.wikipedia.org/wiki/Integration_testing) *[Accessed 20th November 2009]*

*Software Quality Assurance: from theory to implementation,* Daniel Galin, Pearson Education, 2004

*Top-down and Bottom-up, Wikipedia [online], Available at:*

[*http://en.wikipedia.org/wiki/Top-down\_and\_bottom-up\_design*](http://en.wikipedia.org/wiki/Top-down_and_bottom-up_design) *[Accessed 20th November 2009]*

*Handbook of Software Quality Assurance,* Gordon Schulmeyer, Artech House, 2007

*System Testing, Wikipedia [online], Available at:*

[*http://en.wikipedia.org/wiki/System\_testing*](http://en.wikipedia.org/wiki/System_testing)*[Accessed 20th November 2009]*

*Software Development Lifecycle blog, Joe Young [online], Available at:*

[*http://what-is-sdlc.blogspot.com/*](http://what-is-sdlc.blogspot.com/) *[Accessed 20th November 2009]*

1. Integration testing, Wikipedia*:* [*http://en.wikipedia.org/wiki/Integration\_testing*](http://en.wikipedia.org/wiki/Integration_testing) [↑](#footnote-ref-1)